

IndustrialBrainMCP Pro

Installations- und Benutzerhandbuch für das native Linux-Speichersystem

1. Einleitung: Sinn, Zweck & Nutzen

IndustrialBrainMCP Pro ist ein kompromisslos robustes, für den industriellen und professionellen Dauereinsatz konzipiertes Gedächtnissystem für Large Language Models (LLMs). Es schließt die Lücke zwischen experimentellen Prototypen und ausfallsicherer Systemarchitektur.

Der **Sinn** liegt in der dauerhaften, fehlerresistenten Speicherung von Kontext, Wissen und Interaktionshistorien von KI-Modellen, ohne auf fehleranfällige Container-Virtualisierung (wie Docker) angewiesen zu sein.

Der **Zweck** besteht in der Bereitstellung einer nativen Linux-Architektur, die auf standardisierte Protokolle (Model Context Protocol, MCP) setzt und Transaktionssicherheit (ACID) garantiert.

Der **Nutzen** für den Anwender: Absolute Datenintegrität durch SQLite im WAL-Modus, nahtlose Hintergrund-Synchronisation zu Neo4j und Markdown via systemd, sowie blitzschnelle Integration in Frontends wie LM Studio und Ollama.

2. Kernarchitektur & Technische Merkmale

- **ACID-konforme SQLite-Storage:** Transaktionssichere Speicherung aller Ereignisse im Write-Ahead Logging (WAL) Modus verhindert Datenverlust bei Systemabstürzen.
- **Nativer Systemd Service:** Der Hintergrund-Archivar läuft als stabiler Linux-Dienst (`industrial-archivist.service`) mit automatischem Neustart.
- **Pydantic & Instructor:** Strikte Datentypisierung und strukturierte LLM-Ausgaben garantieren fehlerfreie Datenstrukturen im Vektor- und Graphen-Speicher.
- **Dualer Sync (Neo4j & Markdown):** Automatisierte Spiegelung der Memory-Inhalte in eine Neo4j-Graphdatenbank und lesbare Markdown-Dateien im Dateisystem.

3. Installations- und Setup-Anleitung

Schritt 1: Verzeichnisstruktur prüfen

Das Projekt befindet sich unter `/home/neo/IndustrialBrainMCP_pro` und umfasst die Module `storage.py`, `background_archivist_pro.py`, `mcp_server_pro.py` und `config.json`.

Schritt 2: Systemd-Dienst einrichten

Die Service-Datei unter `/etc/systemd/system/industrial-archivist.service` steuert den Hintergrundprozess:

```
[Unit]
Description=IndustrialBrainMCP Pro Background Archivist
After=network.target neo4j.service ollama.service

[Service]
Type=simple
User=neo
WorkingDirectory=/home/neo/IndustrialBrainMCP_pro
ExecStart=/usr/bin/python3 /home/neo/IndustrialBrainMCP_pro/background_archivist_p
Restart=always
RestartSec=10

[Install]
WantedBy=multi-user.target
```

Dienst aktivieren und starten:

```
sudo systemctl daemon-reload
sudo systemctl enable industrial-archivist.service
sudo systemctl start industrial-archivist.service
```

4. Integration in LM Studio (mcp.json)

Ergänze deine Konfiguration in LM Studio um den neuen Server-Eintrag:

```
{
  "mcpServers": {
    "memory-system": {
      "command": "/home/neo/brain1/venv/bin/python",
      "args": ["/home/neo/brain1/mcp_server.py"],
      "env": {
        "NEO4J_URI": "bolt://localhost:7687",
        "NEO4J_USER": "neo4j",
        "NEO4J_PASSWORD": "000000000",

```

```
        "MCP_MEMORY_BASE_DIR": "/home/neo/brain1/memory_system"
    },
    "IndustrialBrainMCP_plus": {
        "command": "/usr/bin/python3",
        "args": ["/home/neo/brain2/IndustrialBrainMCP_plus/mcp_server.py"],
        "env": {
            "NEO4J_URI": "bolt://localhost:7687",
            "NEO4J_USER": "neo4j",
            "NEO4J_PASSWORD": "000000000",
            "MCP_MEMORY_BASE_DIR": "/home/neo/brain2/IndustrialBrainMCP_plus/memory_sy
        }
    },
    "industrial-brain-pro": {
        "command": "/usr/bin/python3",
        "args": [
            "/home/neo/IndustrialBrainMCP_pro/mcp_server_pro.py"
        ]
    }
}
```

5. Befehls- und Statusübersicht

Befehl / Aktion	Beschreibung
<code>sudo systemctl status industrial-archivist</code>	Prüft den laufenden Status des Hintergrund-Archivars.
<code>sudo systemctl restart industrial-archivist</code>	Startet den Dienst nach Code-Änderungen neu.
<code>journalctl -u industrial-archivist -f</code>	Live-Ansicht der Logausgaben zur Fehlersuche.